DenseNet on CIFAR10

Pablo Ruiz – Harvard University – September 2018

Introduction

This work is a continuation of the previous tutorial, where we demystified the DenseNet following the original paper [1]. However, this structure is built to perform well on ImageNet dataset.

ImageNet dataset consist on a set of images (the authors used 1.28 million training images, 50k validation images and 100k test images) of size (224x224) belonging to 1000 different classes. However, CIFAR10 consist on a different set of images (45k training images, 5k validation images and 10k testing images) distributed into just 10 different classes.

Because the sizes of the input volumes (images) are completely different, it is easy to think that the same structure will not be suitable to train on this dataset. We cannot perform the same reductions on the dataset without having dimensionality mismatches.

We are going to follow the solution the authors give (to DenseNets to train on CIFAR10 to build the DenseNet-BC with 100 layers and growth factor of 12), which are also tricky to follow like for ImageNet dataset. On the paper [1], section ***3 DenseNets – Implementations details***, the configurations to build a DenseNet for CIFAR10 are provided.

Let's follow then the literal explanation they give to construct the DenseNets. Most important of all, **for CIFAR10 implementation there are 3 DenseBlocks**, instead of four, **with equally distributed number of DenseLayers in each DenseBlock.** Because the parameter given by the authors is the total number of layers of the network L, we compute how many dense layers within each dense block we need to include to reach that configuration. Being θ the compression factor:

$$nDL = \begin{cases} \lfloor (L-4)/3 \rfloor, & if\ \theta = 0 \\ 0.5 * \lfloor (L-4)/3 \rfloor, & if\ \theta = 0.5 \end{cases}$$

The reason to remove 4 layers is because we need to consider only those belonging to dense blocks. And also, the compression is introducing a new operation in the transition layer (the first 1x1 convolution as we saw in the original work)

Therefore, for a 100 layers DenseNet-BC, each DenseBlock will consist on 16 Dense Layers. Since we will be calling the DenseNets following "DenseNet-Layers-GrowthRate", the DenseNet covered in this work corresponds to DenseNet-100-12.

Structure

Following the same methodology of the previous work on DenseNets, let's take a look at the overall picture first, to go into the details layer by layer later.

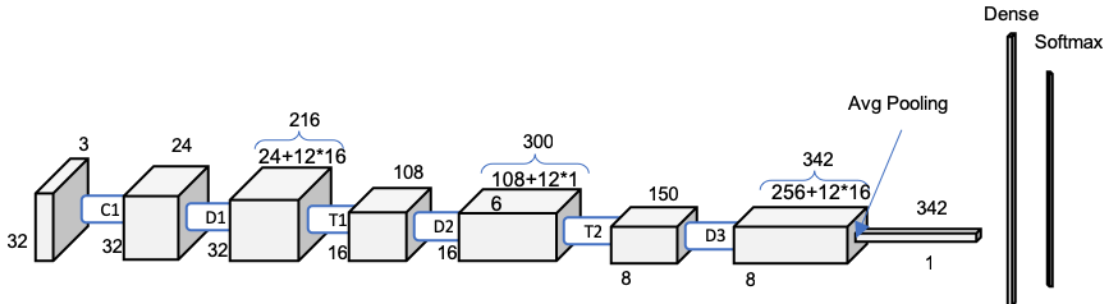

Figure 1. Scheme DenseNet-100-12 on CIFAR10

Figure 1 looks already familiar after demystifying ResNet-121. We can observe the same pattern, a first single convolutional layer, followed by two pairs of dense blocks – transition block pairs, a third dense block followed by the global average pooling to reduce it to the 1x1x342 vector that will feed the dense layer.

Convolution 1

The first step on the DenseNet before entering into the first Dense Block is a 3x3 convolution with a batch normalization operation. The stride is 1 and there is a padding of 1 to match the output size with the input size. Note how we have already our first big difference with DenseNet for ImageNet, that we have not include here the max pooling operation in this first block to reduce the dimension of the input volume.

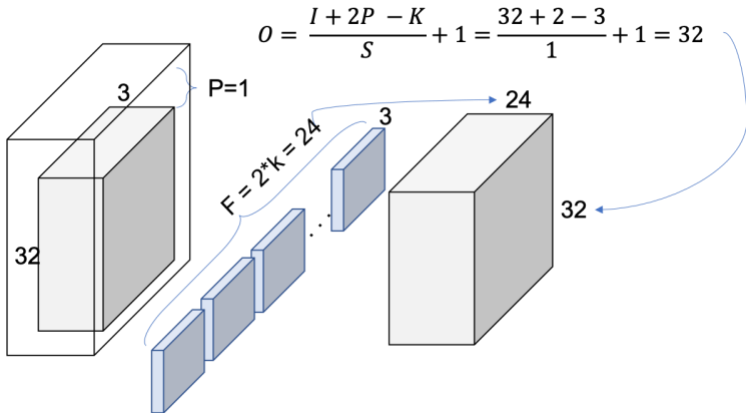

Figure 2. Conv1

We can check with Figure 1 that the output volume of Conv1 is indeed 32x32x24. The next step is to introduce this new volume as input to the following dense block 1 (D1).

Dense Blocks

The dense blocks (DB) are constituted by 16 dense layers as calculated previously for DenseNet-BC-100-12.

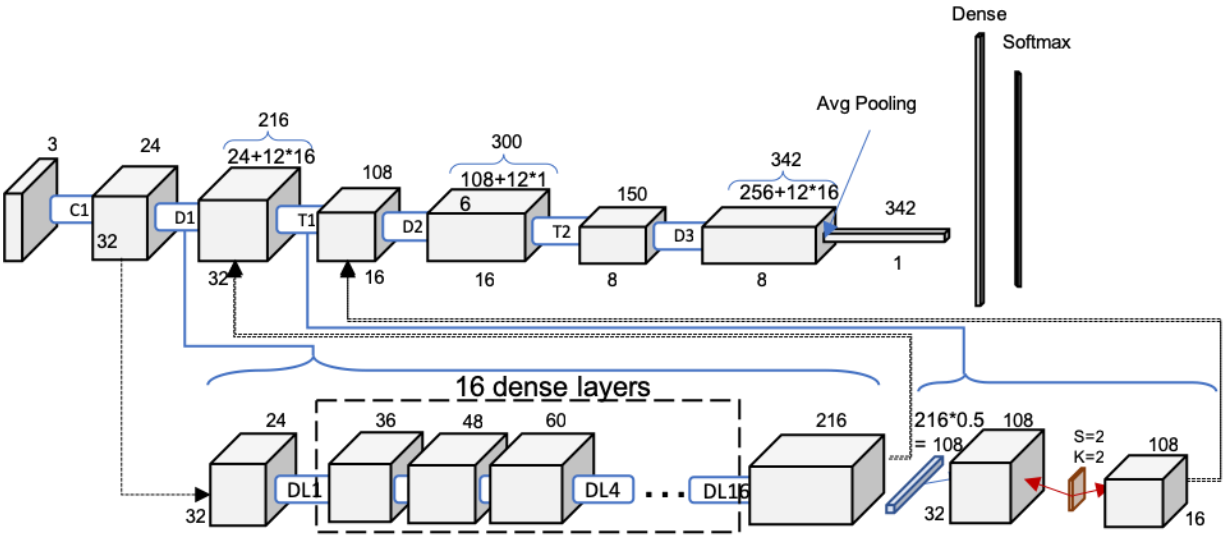

Figure 3. One level deeper look at Dense-100-12. Dense Block and Transition Block. DLx: Dense Layer x

Figure 3 represent a simplified version since 16 blocks where too much to compress in one image, but the idea is followed as we could see how the number of feature maps increase by the growth rate, 12, each time (24, 36, 48 …). Since there are 16 layers, the final volume of the DB1 will be 216.

Transition Blocks

The Transition Block (TB) in between two DBs acts as a down sample, reducing in half the number of feature maps (theta = 0.5) and also reducing in half the feature maps size by pooling with stride = 2, a kernel size = 2 and a padding = 1.

Dense Layers

We just need to see what is happening within a DB to confirm why the feature maps size increase the (growth rate * the number of dense layers) / 2 – Check how in D1 increases from 24 to 216 – and the size of the feature maps remains constant.

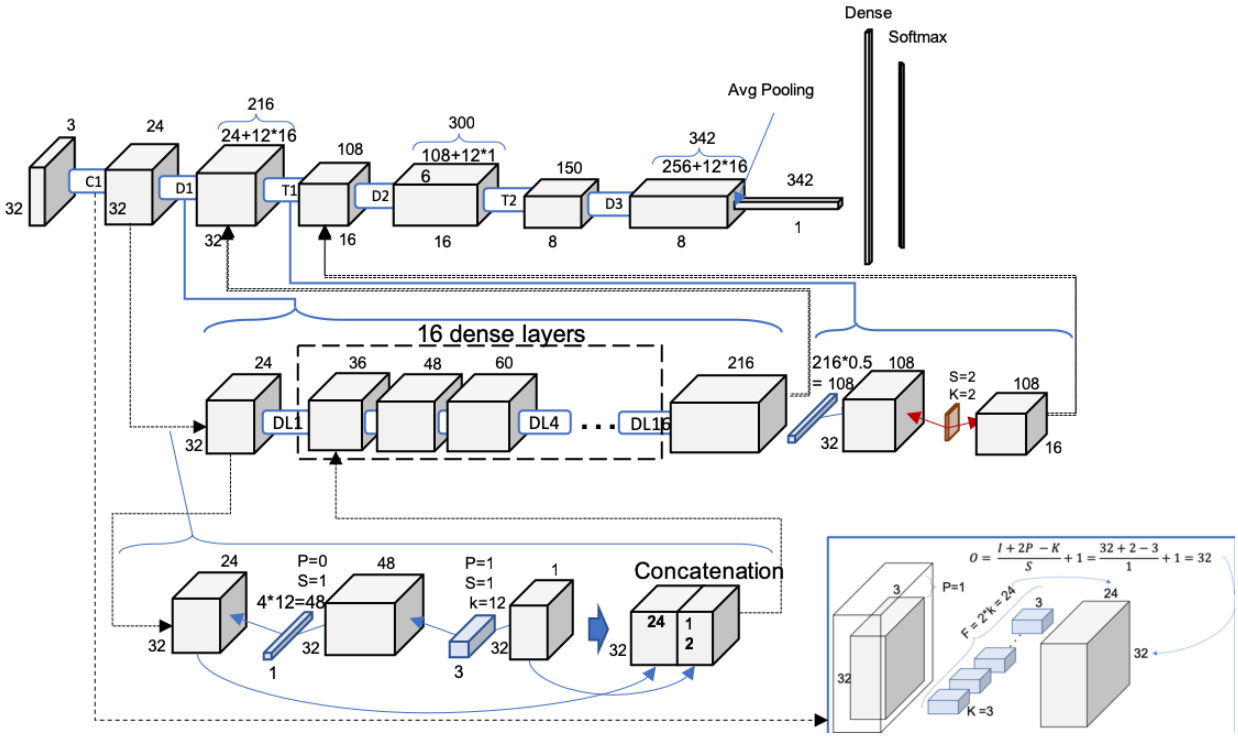

Figure 4.2nd level deep. Full schematic representation of DenseNet-BC-100-12

It is possible to check how each Dense Layer (DL) is performing a 1x1 convolution with 4*growth rate the number of filters. The number 4 is given by the authors in the paper and most of the repositories call it bn_size (multiplicative factor for number of bottle neck layers). This 1x1 convolution is applying a linear transformation previous to the 3x3 convolution with the number of filters being the growth rate.

See how the second convolution is the only responsible for the number of filter maps being concatenated, so it perfectly matches the growth rate configuration after concatenating them. Every DL is adding then k new feature maps to their inputs volume. This reconfirms why a dense block adds growth rate * number of dense layers.

Summary

The rest of DenseNets following the explained rules built by the authors yield to the following structures:

Table 1. ResNets architectures for CIFAR-10

Model	Growth Rate	Layers	Number of Parameters
DenseNet	12	40	1.02M
DenseNet	12	100	6.98M
DenseNet	24	100	27.249M
DenseNet-BC	12	100	0.769M
DenseNet-BC	24	250	15.324M
DenseNet-BC	40	190	25.624M

Note that, intuitively, these architectures do not match the architectures for ImageNet showed at the end of the work on ImageNet.

Find here the code to build these architectures: